Embedded System for Security Services Integration

Edward David Moreno
Federal University of Sergipe (UFS), Brazil
edwdavid@gmail.com

Fabio Dacencio Pereira
UNIVEM and University of Sao Paulo (USP), Brazil
fabioist@gmail.com

Abstract

There are several tools and techniques that can inhibit different and malicious attacks. However, the incorrect use and the lack of data exchange among these security applications can create vulnerability points. In this paper, we propose an Embedded System for Security Services Integration (SSI) and discuss aspects about its architecture and performance when it is implemented in FPGA.

Keywords*: Embedded System, Security Services Integration, FPGAs, Performance, Hardware Security.*

1. Introduction

Researches on security systems have focused typically on creating new services or improving the performance and reliability of a single technique, algorithm or mechanism. The approaches that aim some integration among several security mechanisms have usually focused on a specific strategic application, since the systematic approach for integrating security systems requires an analysis of relations among different data and strategies.

In this context, we like to intend for describing not only an integration model for security services, but also provide mechanisms to implement it in embedded hardware and multiplatform software, and we will focus on requirements as transparency, performance and productivity of our approach.

The research on security systems integration (SSI) may be noted since decade of 70. The concern in become a security technical set in a single integrated system is of paramount importance, since the possible fragility of a security service may be offset by other.

Commercially, network devices and software of security have the challenge for creating this integration. The greatest difficulty is for establishing a common strategy for innumerable devices, tools and techniques of information security.

The study of this integration is not new. The first research about this integration appeared in 1975, and it was proposed by David Elliott Bell and Len LaPadula and it is known as the Bell-LaPadula model [1]. This model was intended to create classes of access control for the DoD systems. After this pioneer model, many techniques to integrate security systems were developed [1-10].

The current models of SSI determine the relationship between a specific set of security mechanisms which exchange information for preventing or treat system's anomaly. So, the most of these models propose solutions under a specific set of services disregarding the existence of other [4-8]. The most of them used UML techniques to represent the features, functionality and methodology.

So, this paper proposes a special architecture and describes the functionalities of an Embedded System for SSI. It was implemented in FPGA of the Virtex family. The different modules dedicated to security as firewall, IDS, proprietary applications, among others, were implemented in embedded hardware.

This work shows an SSI Layer (called as ISSL), which may enclose proprietary security services, opensource solutions, up to applications that provide security as a necessity or a plus. The ISSL has a common structure capable to combine the security services that belong to a particular computational system. The information stored in the ISSL can be analyzed using behavioral models and generate graphs that represent the behavior of different attacks. These behavioral models can be efficiently used to detect early attacks, reduce false positives, classify the severity of the attack and define the techniques used in the defense, among other advantages that are described in this paper.

The Modified Hidden Markov Model (MHMM) was used to determine the behavior of malicious attacks. The MHMM was applied in ISSL and allowed the implementation of an efficient protection system that uses the integration of security services as predominant factor.

2. Background and Related Works

The integration of security services is a necessity since the attacks techniques are using the integration to increase the strength and efficiency of the attacks. Some works have solutions for integrating a specific set for security services set and they are bringing protection to the system.

The model proposed by Nimal Nissanke [8] focused on the protection of three pillars: secret information, user identification and access control mechanisms. The current models have mainly focused on services integration of access control and intrusion detection, they created a prevention system and dynamic protection.

Kim, in his work [11], affirms that Conventional security systems provide the functions like intrusion detection, intrusion prevention and VPN individually, leading to management inconvenience and high cost. To solve these problems, attention has been paid on the integrated security engine integrating and providing intrusion detection, intrusion prevention and VPN.

The main concept discussed by Zilys, in his paper [5], involves representation through objects graphs that represent security services. From this concept, may be a link among security events. The concept, which is formulated, enables strategic control of integrated security systems (ISS) considering from the influence-reaction parameter point. Reaction strategy algorithm selection allows a minimizing of reaction time to danger influence and a maximizing efficiency of security system. The author does not explore the different possibilities of constructing security objects. The paper presents only a basic form that could be further explored.

Jonsson [7] proposed a model that defines security and dependability characteristics in terms of a system's interaction with its environment via the system boundaries

and attempts to clarify the relation between malicious environmental influence, e.g. attacks, and the service delivered by the system. The model is intended to help reasoning about security and dependability and to provide an overall means for finding and applying fundamental defense mechanisms. Since the model is high-level and conceptual it must be interpreted into each specific sub-area of security/dependability to be practically useful.

The model proposed by Jonsson is suggested for an integrated conceptual model of security and dependability. The model is aimed for improving the understanding of the basic concepts and their interrelation. This model categorizes system attributes into input, internal and output attributes, whether from the traditional security domain or the traditional dependability domain. It should be helpful for reasoning about security, so that effective defense methods can be developed and tangible results with respect to security/dependability performance can result. Furthermore, the model is intended to be used for the development of security metrics. We did not find this model applied in a specific case study.

One more recent proposal of Hassan Rasheed [9] proved the integration of services, such as intrusion detection and access control. The author specifies how to respond to intrusion detection.

Along this same line other researchers have proposed similar works. The differential of our proposed system with those related works is that our approach tries to explore the universe of greater security services and the union of features and functionality in an integration layer.

The main contribution of our work when compared with others listed earlier is that in addition to presenting a different model, it also described mechanisms and architecture capable of linking various security services into a single structure.

Despite of the difficulty in defining a strategy for creating a model of integrated security services, there are good solutions that have satisfactory performance.

The integration of security services is the main objective of this work and here we promote a common data structure to store details of detected anomalies. Thus the information generated by security services in a computer system can be analyzed and used for decision making in order to prevent anomalies.

In this context, the solutions highlighted in the sequence are techniques used to analyze the behavior of computer systems in order to distinguish normal from anomalous situations.

Yasami [13] presents an ARP-based anomaly detection algorithm using Hidden Markov Model in enterprise networks. This method can verify the ARP traffic of a network to create graphs, representing the normal behavior of a computer system. Due to this characteristic, it requires a period of training. The longer of the training period and the better will be the representative and specificity of the model graph. As results, behavior graphs were presented and the results were positive.

In this method, there was no integration with any security service, and the detection system works in a low level of the OSI model, between physical layer and link layer. However, the most important point of this work is that it clearly presents a real application of the Hidden Markov Model Modified (MHMM) for security problems.

One equivalent technique was presented by Shrijit [12], although applied to a higher level of the OSI model, between the transport layer and network. This work does not present in detail the impact of intrusion detection.

Close to this functionality, but already at implementation level, there are intrusion detection systems, known as IDS, and intrusion prevent system (IPS), but they still operate by itself, without integration with others security services.

After this study phase about the available models, an important point of discussion appears: What is the most appropriate level in the OSI model to install an integrated security that looks for efficiency and effectiveness in the protection and prevention of computer systems?

With this focus, the present paper proposes as solution with an Integrated-Services Security Layer (ISSL) that operates between the operating system and applications, where it was created a common structure for integration.

The ISSL consists of an integration model that does not ignore any kind of security service. In other words, it was not designed to replace a specific security service, but to create and compose an integrated and effective model for protecting computer systems.

3. Embedded System for SSI

Our embedded system for Security Services Integration (SSI) can be divided into cores in hardware and libraries in software that collaborate among themselves for supporting the specifications of an integrated security system.

Our proposed system uses security services such as IDS, firewall, antivirus, antispy, and audit aspects. These services can do tasks for cooperating among them through an Integrated-Services Security Layer (ISSL). The main function of the ISSL is unifying the security services so that an application would have access to them through a common database and a specific set of methods. The advantages can be summarized mainly by transparency access to security data, productivity, and robustness and achieved performance.

The technology and methodology adopted for developing this project have direct impact on the viability, power consumption, area, complexity, flexibility and other factors related to the final application. In this context, we have chosen hybrid architecture.

This model supports the integration into a single system, hardware and embedded software. So, by using a single interface for communication could be made that hardware and software have interactions, extracting the benefits offered by both.

For classifying the services that would be respectively implemented in hardware and software we have adopted the following methodology. In first version, the system was described in embedded software and it is executed by the PowerPC processor which is embedded in a FPGA Virtex.

After, we have created some libraries with the function of time calculation and instructions counting (timer.h and InstC.h). These were used to detect code sentence and instructions sets that consume high processing time.

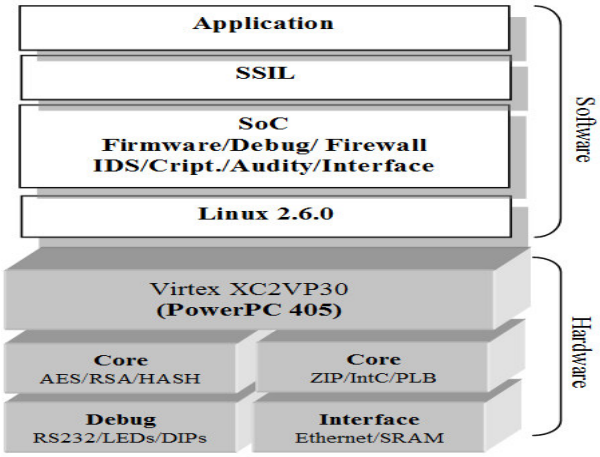

Figure 1. Hardware and Software of our Embedded System

Based on these results optimizations were performed at the level of software, but the main contribution was the competence for detecting the system functions that could be implemented in hardware. In the sequence, we present the organization in layers of our embedded system.

The final architecture is shown in figure 1 and it is composed of dedicated cores and embedded software that implements the system's control. The software runs under the Linux operating system version 2.6.0, and this distribution was dedicated and compiled for the PowerPC processor 405.

Figure 1 describes the organization of our embedded system. As can be seen in Fig. 1, the system's features are divided into two categories: (i) high performance functions which are implemented in dedicated hardware (cores) and (ii) control functions, data structures and flow process, that are implemented in embedded software. In this case the C language was adopted as the best performance to native applications for PowerPC processor.

The PowerPC processor physically present into the FPGA Virtex is responsible for running the embedded software including the security functions, flow control and operating system installed, which were also implemented in this project.

The project's architecture is composed of dedicated cores that communicate among them with the PowerPC processor through controlled buses (PLB). Fig. 2 shows how the embedded system proposed is organized, highlighting the main cores and software stored in memory of 512 Mbytes RAM.

We have projected a preliminary version of this system by using the XUPV2P platform which consists of a FPGA Virtex 2 Pro (XCV2P30) and peripherals of Input and Output, where there are the interfaces used as Ethernet, memory SRAM, RS232, Leds and Switches.

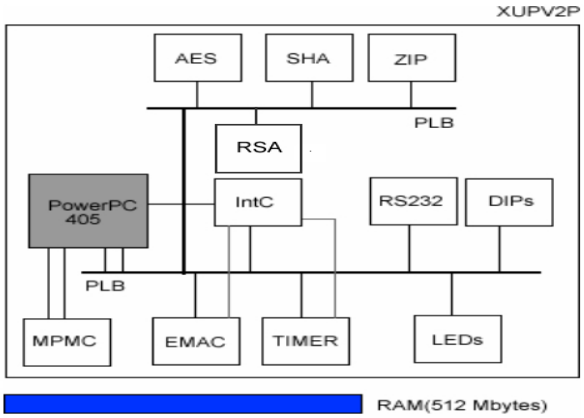

Figure 2**.** Top-level Architecture of our Embedded Hardware

3.1. ISSL - Layer of Integrated-Services Security

The Integrated-Services Security Layer (ISSL) describes a mechanism to store information about security system, in pursuit of different devices, techniques and tools share relevant information. The ISSL will attend a high number of devices, tools and algorithms that exist and that still will be created. This is possible only if have a dynamic structure that allows customize its characteristics, maintaining the integrity of all information which is can be stored locally for each security service, and this can generate an event and register in ISSL frame (see Figure 3). All and any anomaly of system identified by some of security services must be formatted according as structure shown in each frame.

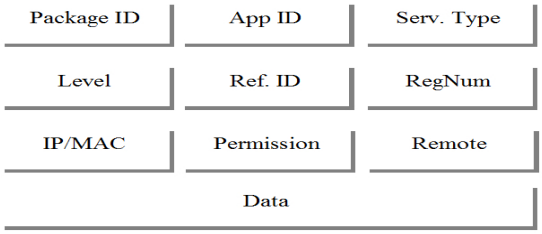

Figure 3 – Structure for SSI using a ISSL frame

This structure contains sufficient information to a security policy adopted a decision consistent in regarding indications of possible anomalies. This structure aims to prioritize the network access control. That is, the sum of deficiencies identified by the system can lead to blockage of access. Thus, devices of a network that make anomalies will be blocked. The system will be able to deny access to network without necessarily knowing where or what security services identified the anomaly.

The storage structure (frame) is shown in Fig. 3 and is described in the sequence:

- Package ID: identifies a single package; field with auto-increment.
- Application ID (2 bytes): This field stores the application ID that generated the notification.

- Service Type (1 byte): classifies the security service type that detected the anomaly.
- Anomaly Level: quantifies and specifies the depth of anomaly.
- Reference ID: reporting reference number (more details on anomaly identified).
- Register Number (1 byte): sequential number indicating the number of notifications made by a particular application on an identified anomaly.
- IP / MAC (10 bytes): register the IP number and MAC address of the creator of notification
- Permission (4 bits): Set the access level
- Remote access (4 bits): allows or not remote access to notification.
- Data (100 bytes): information about notification.

The Decision-making after the identification of an anomaly is the responsibility of the application that uses the ISSL. For the initial tests was adopted a simplified model of integrated services based on the security structure published in 2007 by Nissanke [8].
Specific information about any anomaly may be necessary for the decision. This information can be accessed through the Reference ID field in our data structure that has the function to point the index or code for locating the anomaly generated by a specific security service.
The ISSL architecture from a point of view top-down can be divided in four main modules:

- Storage structure: composed of related tables capable of store the information about anomalies.
- Behavioral Models Builder: based on the Modified Hidden Markov Model, the system can create a knowledge base through of training period, and by importing detection rules or creating direct behavioral models.
- Anomaly detection system: based on anomalous behavior models, the system identifies attacks in normal execution mode and run-time execcution.
- Decision-Making System: once detected the anomaly, this module follows the action rules of protection and prevention system.

The information will be generated by security services in the system, such as firewall, IDS, antivirus, anti-spyware, integrity mechanisms and others. This information will be evaluated by ISSL with the ability to analyze and consider possible actions and protect the system.
In this context, the ISSL operates more broadly than a simple IDS, because its knowledge base for decision making is composed by several sources (different security services), which become more effective the protection system. To summarize, the main objectives of the ISSL are:

- Reduce the incidence of false positives;
- Allow the creation of precise action rules against identified attacks;
- Allow the early identification of new attacks;
- Integrate different and available security services;
- Allow a better visualization of the attack's behavior.

3.2 Behavioral Models

The behavioral models are used for classifying normal and abnormal activities on a computer system. The behavioral models can be applied for finding sequences of anomalies detected by our ISSL. Among the techniques mentioned, Hidden Markov Model (HMM) was used to formalize the sequences of anomalies that may characterize an attack on a computer system.
A Hidden Markov Model (HMM) is a statistical model where the system being modeled is assumed to be a Markov process with a finite set of states and unknown parameters, each of the states is associated with a (generally multidimensional) probability distribution. It generates internal and external sequences of symbols using probabilistic rules and the challenge is to determine the hidden parameters from the observable parameters.
Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. It is only the outcome, since it is not one state visible to an external observer and therefore states are "hidden" to the outside. This is what the name Hidden Markov Model comes from. HMMs are not exhaustively treated in this paper; has details about HMM.
The direct model allows the system administrator to build detection and action rules conformed his knowledge and experience on anomalies detection. This model may induce the creation of inconsistent rules and that can disrupt the normal functioning of the system and does not protect from the risk situation. However, the direct model allows the customization of security system to a particular computer system.
The ISSL allows the automatic creation of behavioral models for anomaly detection using HMM technique, even as the creation of direct models.
The motivation for choosing the HMM as anomaly detection mechanism can be justified by following features found in our ISSL:

- Finite states number: the states that compose the HMM are unique. Thus, how many more states there are more complex is the transition graph and further results analysis. For ISSL the states are associated with the origin IP address and attack's characteristics. Always the model is pointing to a single state.
- The transitions between states occur by events. For our ISSL, any event means a recurrence detected by security services in the system, in other words, the transition of state occurs when new occurrences of anomalies arise.
- Building models for abnormal conditions: the project developed the MHMM and builds graphs of anomalous conditions of the system. Thus, they can be generated for robust signatures of attacks that can be distributed and used in other computer systems.

The HMM model can be modified to suit characteristics with respect to one important item, the impact human behavior on the period of training and creation of a model of behavior.

Yasami [13] modified the HMM and applied their new equations for creating models of normal behavior of an intrusion detection system based on ARP requests, and his results were satisfactory. This work was used in the modified HMM (MHMM) [13]. But for generating models of abnormal behavior, it is possible create a robust characterization of attacks and provide important information to decision-making system which can acts on a specific attack with more efficiency and effectiveness.

The ISSL allows a continuous training, in others words, the system in real time can feed the MHMM as new states and transitions. Therefore, the model can be adapted in runtime to new attacks or new applications installed in the computer system.

The impact on processing is directly related to modules of security services that are active and type of attack that is being done. We have implemented three different version of our approach: software by using Java language, hardware by using a FPGA-based SoC (System On-Chip) and one special simulator.

4. Tests and Results

In this section will discuss the impact of the creation of model behavior and system performance. The detection methods and some security services were implemented on embedded system and submitted to a real test environment.

The test environment is composed by the embedded security device (security services + ISSL) and a test circuit that generates artificial sequences of attacks. The details of attacks made and detected, as well as the rules of protection used in real time are stored in a RAM memory in the embedded hardware board.

The information after end of training and information system in normal mode operation will be discussed below. In our tests, the training period was 5 days, where attempts of the malicious attacks have been made in specific points in network. The security services created were: AntiSpy, Antivirus, IDS, Firewall and two proprietary applications, one is Webserver LITE and other is user authentic system.

Following the information summary of the proposed scenario, our data results are presented in Figures 4 and 5 which correspond to anomalies detected and processed on our embedded system.

The Figure 4 represents the number of states created after training period and it is possible to observe that the number of states tends to stabilize even with the substantial increase in the number of attacks. This feature proves that the choice of MHMM was appropriate for the characteristics of the environment.

The Figure 5 represents the growth in the number of transitions between states as the attacks trend in training period. There is an evolution similar to the number of states and also tends to be stabilized. The occurrence of new attacks can only create new states or new transitions.

The order of attacks generation can change the order creation of states, but the end result gives equivalent models in relation to behavior. After the training period the behavior graphs generated were inserted into the actual implementation of the ISSL.

During a month, there were daily attempts of attack on real server. The attacks were realized by tests devices. In this period of normal operation, important data were generated. Among these highlight unidentified attacks that hit 4% of attacks realized.

New attacks were implemented in the real environment, even as the ISSL was partially able to recognize the attempts of further attacks and enable the decision-making system.

This result was expected, because does not provide a sufficient number of attacks to create a robust MHMM in training period. Each attack not foreseen in MHMM generated a log record, it happens every time that a new state or transition occurs and is not foreseen in the model generated in training period. The number of false positives was reduced to a rate below 0.5%, where the sensitivity of the system can be considered in characterization of the detection rules.

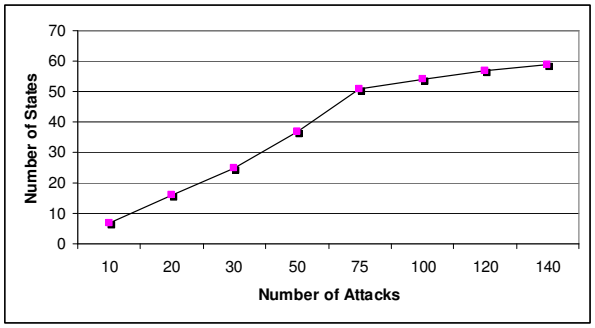

Figure 4 - Number of states created (MHMM)

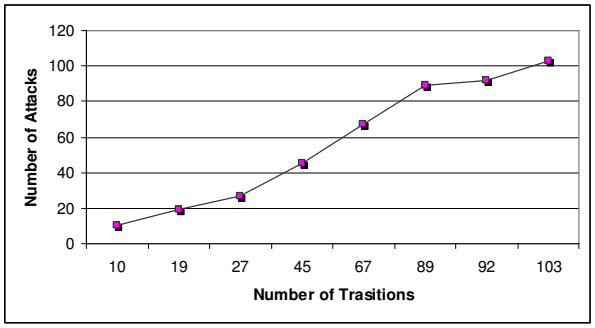

Figure 5. Number of transitions created (MHMM)

One important factor is that our ISSL accurately detect different attempts of attacks and frustrated notified and also promote the protection and prevention of computer system, specifically attending the identified deficiencies.

Attacks of similar nature have been eradicated by their own security applications after the action of decision-making system.

The impact on processing is directly related to implemented modules to each security services which are active and type of attack that is being done. In our test, we use four (4) different situations: full attacks, attacks varied by sequence, attacks varied by time, and finally, when they are a mix between time and sequence. The last situation (hybrid of

time and sequence) is more random, but, in spite of random our models can detect the anomalies. Sequential-type attacks (burst) can stress the system and raise the utilization rate of dedicated process modules and embedded processor.

It is possible note that we use different techniques for security: firewall, IDS, applications proprietary, antivirus and antispy; so we have a real scenario of integration of security services. To improve the performance, we inserted other processor; so a second embedded processor in the Virtex FPGA was activated, then dividing the tasks of processing. Thus the architecture model for this version has two hardcore processors actives.

Another modification was made in the embedded software since we improve the implementation of our ISSL routines. In this case it was possible the use of two processors, since many tasks are independent, allowing the implementation in parallel.

The average rate of use was calculated using the number of calls to a particular module and the execution time for each routine called. The average load related to utilization of processor hidden the overload time of the system. With a single embedded processor, we note that the usage of processor is of 87% and it remains with maximum load for 25% of the total time examined. When we have 2 processors this rate was reduced to 4%, since the usage of each processor is around 55%.

5. Conclusions

This work presented a specialized Embedded System of Integrated Security Services. We propose and discuss the architecture, features and performance when it is projected in a FPGA Virtex. The security services were created and we have a prototype of the embedded platform which offers integration among different security services. Our embedded system offers transparent and user-friendly provided to the user, productivity and greater involvement of the security mechanisms of the present system.

The Integration-Security Services Layer allowed to created links between all the security services of the proposed scenario. The common information structure allowed to storage fundamental data for creating behavior models of attacks and the integration with the action rules. The prevention mechanism has settings specific to the security services inhibit the identified attacks in the behavioral model.

Acknowledgements

The authors would be like to thank the FAPITEC/SE (Fundação de Amparo à Pesquisa e à Inovação Tecnológica do Estado de Sergipe) for financial support under the grant No. 019.203.01185/2010-2.

6. References

[1] Bell, D. E Lapadula L. “Secure Computer System: Unified Exposition and Multics Interpretation”. Technical Report MTR-2997 Rev. 1, MITRE Corporation, Bedford, MA, 1975.

[2] Baker, M.P, “Integrated security system”, Proceedings International Carnahan Conference on Security Technology, 1989.

[3] Okamoto, E, “Proposal for integrated security systems”, Proceedings of the Second International Conference on Systems Integration ICSI '92, 1992.

[4] Ferraiolo, D. F.; Sandhu, R.; Gavrila, S.; Kuhn, D. R.; Chandramouli, R. “Proposed NIST standard for role-based access control”. ACM Transactions on Information and System Security, v. 4, n. 3, p. 224-274, ago. 2001.

[5] Zilys, M.; Valinevicius, A.; Eidukas, D.,”Optimizing strategic control of integrated security systems”, 26th International Conference on Information Technology Interfaces, 2004.

[6] Ghindici, D.; Grimaud, G.; Simplot-Ryl, I.; Yanguo Liu; Traore, I., “Integrated Security Verification and Validation: Case Study”, IEEE Conference on Local Computer Networks, 2006.

[7] Jonsson, E., “Towards an integrated conceptual model of security and dependability, Availability, Reliability and Security”, 2006. ARES 2006.

[8] Nissanke, N, “An Integrated Security Model for Component–Based Systems”, IEEE Conf. Emerging Technologies & Factory Automation, ETFA, 2007.

[9] Hassan, R. and Randy Y.C., “ChowAn Information Model for Security Integration”, 11th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'07), 2007.

[10] Pereira, F. D., Ordonez, E. D. M. “A Hardware Architecture for Integrated-Security Services”. Transactions on Computational Science – Special issue on Security in Computing, Springer Verlag LNCS 5430, v. 1, p. 215-229, 2009.

[11] Kim, J. Design and Implementation of Integrated Security Engine for Secure networking. In IEEE Advanced Communication Technology, 2004.

[12] Shrijit S. Joshi and Vir V. Phoha. “Investigating Hidden Markov Models Capabilities in Anomaly Detection”, In 43rd ACM Southeast Conference, March 18-20, 2005, Kennesaw, GA, USA.

[13] Y.Yasami M.Farahmand V.Zargari, “An ARP-based Anomaly Detection Algorithm Using Hidden Markov Model in Enterprise Networks”, IEEE Intl. Conference on Systems and Networks Communications (ICSNC), 2007.